Remarks and Arguments

Applicant's attorney wishes to thank the examiner for his time and consideration in a telephone interview conducted on February 6, 2004 during which the outstanding rejection of claims 1-36 and the Atkinson and Koppolu references were discussed. The substance of that interview is set forth below.

Claims 1-36 have been submitted for examination and have been rejected under 35 U.S.C. §103(a) as obvious over U.S. Patent No. 6,263,379 (Atkinson) in view of U.S. Patent No. 6,460,058 (Koppolu.). As discussed in the telephone interview, in the present invention, a moniker object assists in persisting and resurrecting distributed objects so that a life cycle services system, located in local storage, can interact with the distributed objects and provide life cycle services.

In particular, when an object is stored, it is streamed into the local storage. During the streaming of the distributed object into local storage, a moniker object is automatically substituted for the actual object. Thus, the moniker object is stored in local storage in place of the real object. The moniker object in the local storage can then be maintained by the inventive life cycle services system. This operation is described in detail in the present specification at page 22, line 3 to page 23, line 3 (in connection with figures 13 and 14.)

Similarly, when a distributed object is resurrected, the moniker object is instead streamed in from local storage. During this streaming, a reference to the real object is substituted for the moniker object in the memory. The real object is created in the remote server. This operation is described in detail in the present specification at page 23, lines 4-29 (in connection with figures 15 and 16.)

Both the Atkinson and the Koppolu references disclose moniker objects. However, although the objects disclosed in the prior art have the same name as the moniker objects of the present invention, they function in an entirely different manner. Both Atkinson and Koppolu are designed to solve the problem of working with diverse objects, each of which has its own interface. Atkinson discusses this problem in connection with application programs, such as word processing systems, that must operate on compound documents. These compound documents may contain linked or embedded data objects, such as spreadsheet objects, that have diverse interfaces. In the Atkinson system, a moniker object is used to link to the data objects. Different moniker objects can be used to link to different types of objects, such as files, items and pointers. The advantage is that the moniker object has a consistent interface IMoniker. Thus, the application program can interact in a consistent way with the moniker object and the moniker object controls the real objects.

Koppolu uses its moniker objects in a similar fashion, but in the context of a browser in order to manage hyperlinks to non-HTML documents. Again, the moniker object has a consistent interface IMoniker with which the browser operates. Thus, the browser can interact in a consistent way with the moniker object and the moniker object controls the real objects that are hyperlinked.

In both the Atkinson and Koppolu references, the moniker object itself can be stored and retrieved (resurrected.) In addition, in both references, the disclosed moniker objects (and the interface IMoniker) contain methods (BindToObject and BindToStorage) that allow the moniker object to store and retrieve the object to which the moniker object is bound. However, as discussed in the telephone interview, during the storage of the bound object, the moniker object is not substituted for the bound object so that the moniker object is stored in place of the bound object. For example, the moniker object can be used to store its bound object using the BindToStorage method. Then, the moniker object can itself be stored. This, results in both the moniker object and the bound object residing in local storage. If the moniker object was stored in place of the bound object, then only the moniker object would remain in storage after the store operation was complete.

The present claims explicitly recite that the moniker object is substituted for the distributed object in the local storage during the storage operations so that the moniker object is stored in place of the distributed object. For example, claim 1, in lines 7-10, recites "a first stream object which automatically substitutes the moniker object for the distributed object during the streaming of the distributed object out from the memory to the local storage so that the moniker object is stored in the local storage in place of the distributed object." As set forth above, this operation does not occur in either of the cited references. Consequently, amended claim 1 patentably distinguishes over the cited references.

Amended claims 11 and 21 contain parallel limitation to those noted above in amended claim 1 and patentably distinguish over the cited references in the same manner as amended claim 1. Claims 2-10 are dependent, either directly or indirectly, on amended claim 1 and incorporate the limitations thereof. Therefore, they also distinguish over the cited references in the same manner as claim 1.

A similar result is obtained during the retrieval or resurrection of an object. In the Atkinson and Koppolu references, the moniker object must first be loaded into memory. Then, an application program, or browser, can use the moniker object to load the bound object into memory using the BindToObject method. This results in both the moniker object and the bound object residing in memory. In the present invention, the stream reader that loads a distributed object into memory retrieves the moniker object from local storage, but then substitutes a reference to the distributed object for the moniker object during the streaming of the moniker object into memory. This operation results in only the distributed object residing in memory. For example, claim 3, in lines 2-6, recites a second stream object which automatically substitutes a reference to the distributed object for the moniker object during the streaming of the moniker object in from the local storage to the memory so that a reference to the distributed object is created in memory in place of the moniker object. As discussed above neither cited reference discloses such a stream writer. Therefore, dependent claim 3 patentably distinguishes over the cited references. Dependent claims 13 and 23 contain similar limitations.

Claims 12-20 are dependent, either directly or indirectly, on amended claim 11 and incorporate the limitations thereof. Therefore, they also distinguish over the cited references in the same manner as amended claim 11. In addition, these claims also recite limitations that parallel the limitations in claims 2-10 and thereby distinguish over the cited references in the same manner as claims 2-10.

Finally, claims 22-30 are dependent, either directly or indirectly, on amended claim 21 and incorporate the limitations thereof. Therefore, they also distinguish over the cited references in the same manner as amended claim 21. In addition, these claims also recite limitations that parallel the limitations in claims 2-10 and thereby distinguish over the cited references in the same manner as claims 2-10.

Applicant believes the claims are now in allowable condition. A notice of allowance for this application is solicited earnestly. If the examiner has any further questions regarding this amendment, he is invited to call applicant's attorney at the number listed below. The examiner is hereby authorized to charge any fees or credit any balances under 37 CFR §§1.17, and 1.16 to Deposit Account No. 09-0460.

Respectfully submitted

[illegible] Date: 2/11/04

Paul E. Kudirka, Esq. Reg. No. 26,931
KUDIRKA & JOBSE, LLP
Customer Number 021127
Tel: (617) 367-4600 Fax: (617) 367-4656